ACTP
ADVANCED CYBER TRAINING PROGRAM

ADVANCED
Vulnerabilty Research

5-Day course

COURSE DESCRIPTION

The **Advanced Vulnerability Research** course is ACTP's most technical and challenging offering. Designed for those who are looking to take their VR skills to the next level, this five-day course will teach students how to find vulnerabilities using techniques such as fuzzing, symbolic execution, and code property graphs.

Written by a team of vulnerability research experts, this course will teach students how to find bugs such as use-after-free, type co nfusion, memory corruption, and logic bugs in software applications like web browsers and operating system kernels.

Prerequisites: Students are expected to have a firm grasp of x86-64 assembly, strong programming skills in C and C ++, be comfortable with reverse engineering software using IDA and Ghidra.

Relevant Today

MANTECH's expert ACTP instructors teach you the skills required to find unique vulnerabilities in modern software applications.

Inclusive Approach

Builds on foundational knowledge gained from the CNO programmer course to take students to the next level.

Topics Include:

Advanced bug class discovery via fuzzing, symbolic execution, and code property graphs. Also, advanced reverse engineering, deep dives and root cause analysis of modern real world vulnerabilities.

ACTP
ADVANCED CYBER TRAINING PROGRAM

ADVANCED VULNERABILTY RESEARCH

5-Day course

AGENDA COURSE CONTENT

DAY 1

- **Code Auditing:** A thorough review of auditing techniques across source code and binaries. This section gives students the deep technical insights to approach looking for vulnerabilities in open source and closed source applications using advanced code auditing techniques and writing custom analysis scripts for Ghidra.
- **Integer Errors and Race Conditions:** Students will become familiar with identifying undefined integer operations and dangerous synchronization patterns that lead to exploitable bugs.

DAY 2

- **Fuzzing:** Students will learn how to use modern fuzzing tools such as AFL++ to find vulnerabilities in target programs. This includes hands on labs to attack network-based applications.
- **Type Confusion:** A deep dive into understanding and finding type confusion bugs.

DAY 3

- **Advanced Fuzzing:** Building on the previous fuzzing section, students will write custom fuzzing harnesses for hard targets and learn about fuzzing exotic interfaces such as RPC and operating system calls. This section also covers emulation and instrumentation engines such as Unicorn Engine and Frida.
- **Memory Corruption:** This section gives students a deep understanding of the memory corruption bug class and how to detect these flaws during fuzzing.

DAY 4

- **Symbolic Execution:** A deep dive into the cutting-edge VR concepts of symbolic execution from SAT and SMT solvers, intermediate representations and languages, and execution engines. Students will write their own x86-64 symbolic execution engine and learn how to use angr.

DAY 5

- **Code Property Graphs:** An introduction to graph theory will lead students into understanding how to combine abstract syntax trees, control flow graphs, and data flow graphs into code property graphs. From there, modern tools such as Joern will show how these techniques can find subtle vulnerabilities.
- **Use-After-Free and Information Disclosures:** This section details the intricate ways that information disclosures and use-after-free vulnerabilities manifest.

Exam and certification/completion